13th ICCRTS

"C2 for Complex Endeavors"

Employing Service Oriented Architecture technologies to bind a Thousand Ship Navy

CDR Kurt Rothenhaus, USN
Program Executive Office Command
Control, Computers, Communications & Intelligence
Command and Control Program Office
4301 Pacific Highway, San Diego, CA 92110
619-947-5431
kurt.rothenhaus@navy.mil

Dr. James Bret Michael
Professor
Naval Postgraduate School
Monterey, CA
(831) 656-2655
bmichael@nps.edu

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **JUN 2008**	2. REPORT TYPE	3. DATES COVERED **00-00-2008 to 00-00-2008**

4. TITLE AND SUBTITLE **Employing Service Oriented Architecture technologies to bind a Thousand Ship Navy**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Postgraduate School,Operations Research Department,Monterey,CA,93943**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
13th International Command and Control Research and Technology Symposia (ICCRTS 2008), 17-19 Jun 2008, Seattle, WA

14. ABSTRACT
One of the United States Navy?s strategic goals is to foster a ?thousand ship? Navy, by combining the U.S. Navy with partners from around the globe. A critical requirement to join various Navies from such diverse backgrounds and technology levels is a simple, flexible, and economical Command and Control (C2) system that each stakeholder can employ and extend. Web based Service Oriented Architectures (SOA) have potential to provide methods and technologies to aid in combining this force; however SOA technology alone will not deliver the desired software ?out-of-the-box.? To truly reap the benefits of a SOA based coalition C2 system, the U.S. defense acquisition community should host a coalition based international C2 software development site, managed by developers and following the open source industry model. Built on proven industry standards and agreed upon software design patterns, the collaborative development site should encourage the development of scalable and flexible C2 systems to address maritime coalition C2 system requirements. This article will examine some coalition C2 system challenges and how SOA technologies can support interoperability. Additionally, the article will discuss two software design patterns that can aid maritime coalition software development and how those patterns employed in conjunction with a community of interest-open source software development framework can lead to C2 systems to bind a ?thousand ship? Navy.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**20**	

Employing Service Oriented Architecture technologies to bind a Thousand Ship Navy

Abstract

One of the United States Navy's strategic goals is to foster a "thousand ship" Navy, by combining the U.S. Navy with partners from around the globe. A critical requirement to join various Navies from such diverse backgrounds and technology levels is a simple, flexible, and economical Command and Control (C2) system that each stakeholder can employ and extend. Web based Service Oriented Architectures (SOA) have potential to provide methods and technologies to aid in combining this force; however SOA technology alone will not deliver the desired software "out-of-the-box." To truly reap the benefits of a SOA based coalition C2 system, the U.S. defense acquisition community should host a coalition based international C2 software development site, managed by developers and following the open source industry model. Built on proven industry standards and agreed upon software design patterns, the collaborative development site should encourage the development of scalable and flexible C2 systems to address maritime coalition C2 system requirements. This article will examine some coalition C2 system challenges and how SOA technologies can support interoperability. Additionally, the article will discuss two software design patterns that can aid maritime coalition software development and how those patterns employed in conjunction with a community of interest-open source software development framework can lead to C2 systems to bind a "thousand ship" Navy.

Coalition Command and Control Challenges

A well established requirement exists to rapidly establish international maritime coalitions in support of U.S. national policy. These dynamic international coalitions are formed, and changed in support of the Global War on Terror (GWOT), stability operations, and homeland defense – as well as major combat operations. At the Naval War College commencement address of 2006, Admiral Mullen discussed his vision to "...extend the peace through an interconnected community of maritime nations working together."

Creating an international community of sailors has many components, not least of which is a solid mechanism to rapidly create and maintain a coalition of forces that have a shared view of the battle-space, or Common Operating Picture (COP). This COP will involve a widely dispersed, culturally and technologically diverse user group – operating in a flexible but secure environment. Not only should this view of the battle-space provide a graphic representation of blue and red forces, but also should provide a mechanism to allow the commander to clearly direct the force, and provide context and intentions to subordinate commands through a common Command and Control (C2) system. A new, outward focused C2 software system is needed to target these emerging requirements directly linked to the growing number of coalition warfare contingencies, in addition to major combat operations. To effectively develop a solution for a C2 system for the "thousand ship" navy we need to target those requirements that drive a successful coalition and focus on missions those coalitions normally conduct, such as stability operations and Global War on Terrorism operations. Figure 1 describes the relationships between systems and is adapted from the Chief on Naval Operations guidance for 2006 (CNO, 2).

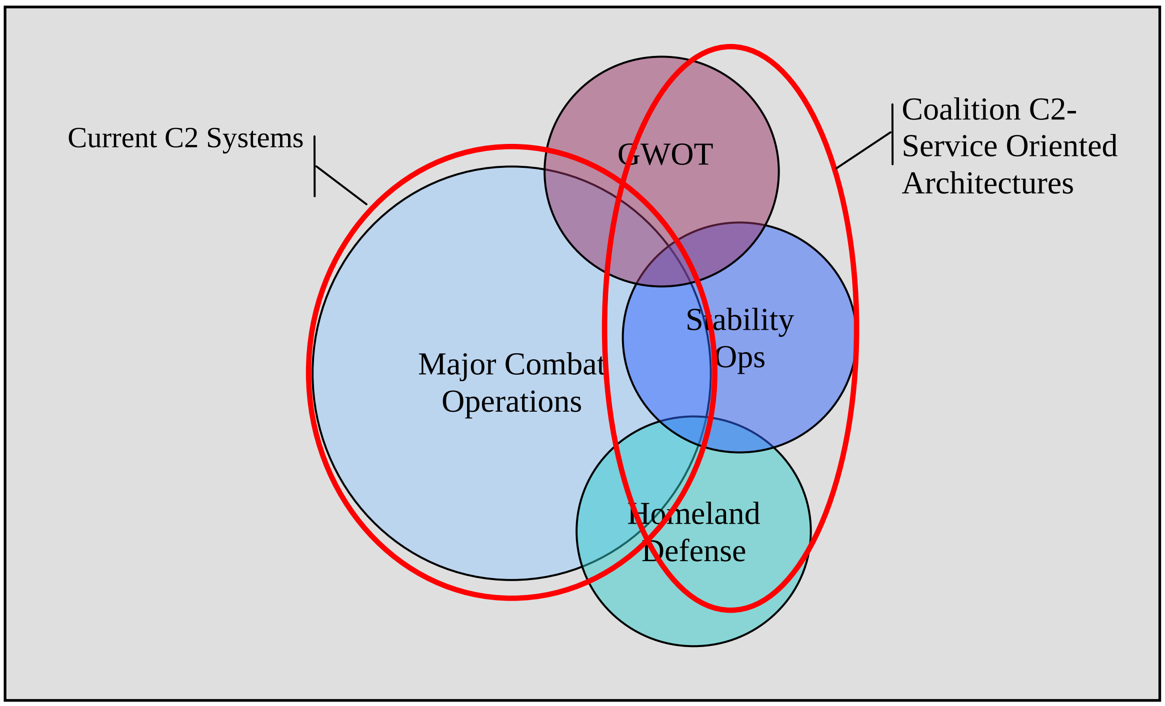


Figure 1. C2 System focus on CNO directed new missions

Numerous operations around the globe are composed of diverse and sometimes large naval coalitions such as: Operation Enduring Freedom (OEF), Iraq Freedom (OIF), and operations in the Pacific and Horn of Africa. These forces assemble to conduct maritime intercept operations as well as anti-piracy and strategic lines-of-communication patrols. These operations, although successful, demonstrated seams in present systems that inhibit us from quickly establishing and disestablishing command relationships between different entities. Present systems also have difficulty in accessing new data types and feeds. For example, commercial blue force position, maritime vessel reporting position data (AIS) or other feeds are slow to enter the Coalition Common Operations Picture and in some cases remain absent. Additionally, the ad-hoc nature of coalition warfare requires flexibility not present in existing C2 systems to connect to new partners, and this is where SOA's provide a potential approaches.

Service Oriented Architectures

Emerging technologies surrounding Service Oriented Architectures (SOAs) offer a compelling tool to solve a number of Common Operational Picture software interoperability challenges. Specifically, SOAs offer:

- **Flexibility**: The ability to rapidly add, delete, and reconfigure COP data sources and modify the COP topology.
- **Cost Savings**: Leveraging commercial investment in technology.
- **DoD investment focus**: Allows the US Navy to focus budgets on military-specific needs.

SOA is a software design framework to decompose larger software programs into smaller, reusable components, or elements of functionality with common, well-defined interface standards (Erl, 2). These elements can then be reused by another program, discovered by other applications, or even rapidly recombined to support changing operations. The US DoD to include the Navy has adopted web-based SOA technologies to address a number of its interoperability challenges and improve efficiency and flexibility (DoD Net-centric, 2). Seeking both cost savings and operational flexibility, the Navy hopes to reduce costs and deliver greater capability. Following a private business IT model of SOA offers the coalition Navy potentially reduced lifecycle costs, greater speed to deployed capability, and the prospect of users being able to reconfigure systems and capabilities to support changing missions. From a C2 perspective, SOA provides the ability to reuse processing logic, such as fusion algorithms, or separate the visualization and presentation tiers of an application from the data management tier, reducing the cost of integration and increasing flexibility.

SOA technology alone will not deliver the C2 system required to build a "thousand-ship" navy as discussed above. Linking a large number of ships within the coalition requires technology beyond what SOA will deliver "out of the box" and also requires a unique acquisition strategy, focused on collaborative development with our partners. This acquisition strategy is a departure from the traditionally centralized acquisition management methods. We should not underestimate the challenges in creating and implementing this new acquisition strategy that is required to design, deliver, and support a SOA based coalition C2 solution. Future systems need to support a broad range of requirements in a shrinking fiscal landscape. A new acquisition strategy for building and integrating these systems relies heavily on our coalition partners and industry-provided foundation technologies, which are collaborative rather than directive and require buy-in from a broad community of stakeholders. This community potentially includes agencies beyond our traditional US Navy partners requiring different development, fielding, and life-cycle maintenance methods. This new approach mirrors industries open source model from the successful Apache and Eclipse development model.

Technical Concepts and Benefits of SOA: Solving the Navy Coalition Interoperability Challenges

A number of articles, books, and tutorials are available discussing the technical underpinnings of SOAs using web-based technologies. Thomas Erl's *Guide to implementing SOA* provides a superb SOA background and clearly describes the benefits and pitfalls of SOA technology (Erl, 5). SOA is a technology agnostic concept however typically it is normally implemented using web-services. Web-services facilitate SOA frameworks by combining a number of web-standard enabled technologies, some of which are mature and others that are not yet widely adopted. The prime technology enabler for SOA is eXtensible Markup Language, which provides the foundation to deliver a number of capabilities, including discovery, orchestration, and data exchange (Erl, 75).

The C2 domain can employ these core capabilities to rapidly integrate new sources of content as well as establish new system relationships in support of Coalition objectives. The vision of SOA is to ultimately allow a user to look into a directory to search for a service and orchestrate a group of services to create further value. Although promising, not all of the technologies are widely adopted and mature. In the case of coalition Common Operations Picture software we can, however, obtain significant gains by employing the more mature technologies such as those in the transport, messaging, and to a limited extent, the description layers of the interoperability stack shown in *Figure 2*, and is adapted from Graham's *Building Web Services with Java* (Graham, 7).

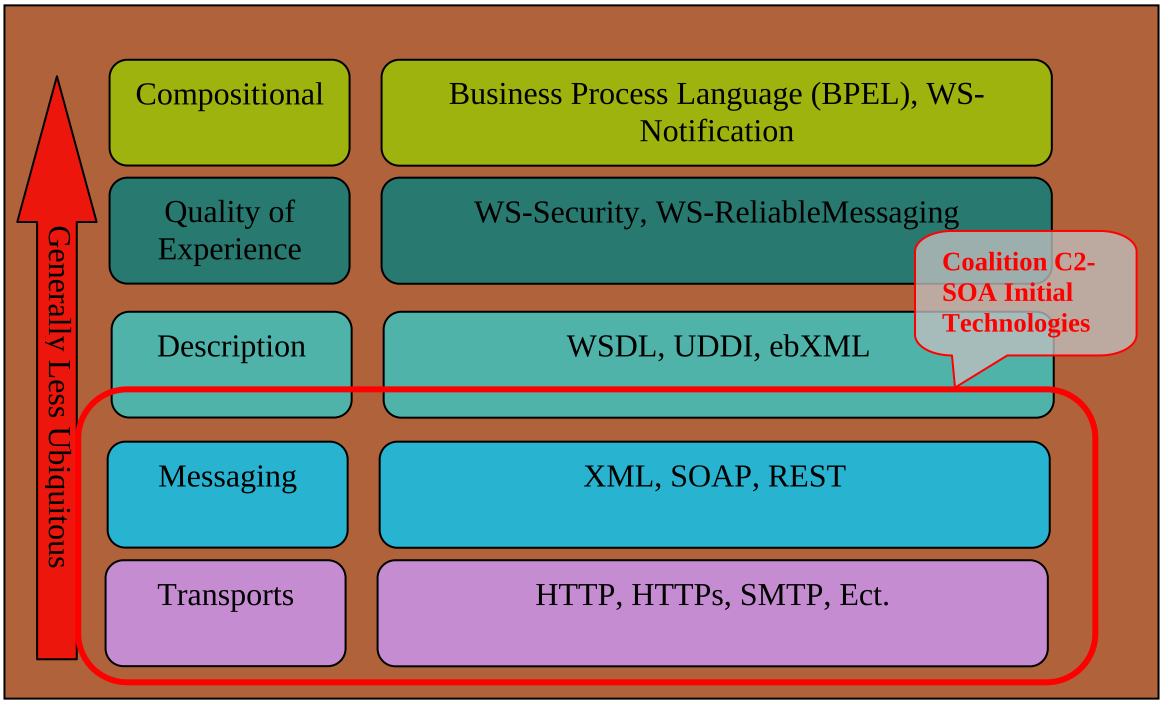
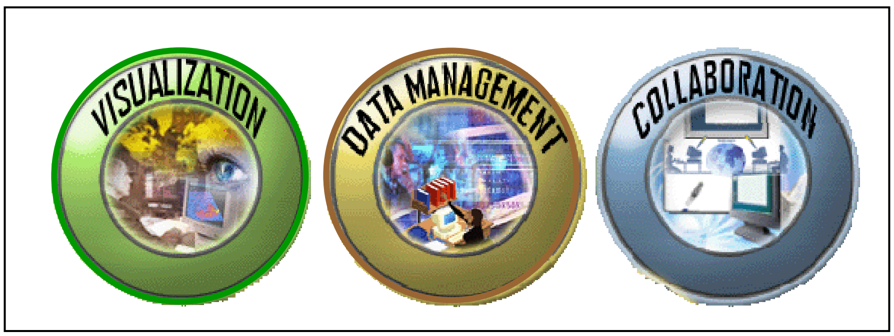


Figure 2. Graham SOA Technology Stack

Examining the interoperability stack further, as we move from bottom to top, in general the technologies at the lower levels are widely adopted by industry, while some of the advanced functionality layers at the top use less mature technologies that have seen limited industry adoption in production systems. For example, HTTP and SMTP are ubiquitous standards and is the foundation of a number of technology solutions. XML, SOAP and WSDL are not yet ubiquitous but are gaining acceptance by industry. The compositional and Quality of Experience types at the top such as business process language are still limited to early adopters. That is not to say we should wait for the entire stack to mature before using the technology. The transport and messaging layers provide interoperability benefits to C2 system design and can be employed today. Utilizing transports and messaging we can separate application logic in COP tracking system logic such as fusion and correlation algorithms to permit reuse – which yields computational and bandwidth efficiency. Additionally, we can add flexibility by lowering the barrier of entry for new data sources and employ more open mapping or presentation systems. These gains allow the DoD to interoperate with a wider audience and leverage commercial investment in IT. Finally in this environment the DoD can focus its resources on those activities unique to the military.

Ad-Hoc Common Operational Picture

In Barnett's book, *The Pentagon's New Map,* he discusses military operations in the "unconnected world" (Barnett, 4). To support the CNO's vision of a "thousand ship" Navy, America needs to reach out to non-traditional partners with systems that can operate in limited computing and bandwidth environments and with existing legacy systems. These future systems should sacrifice functionality for economy and simplicity, but still support three central C2 system requirements of *visualization*, *collaboration* and *data management.* The development of the systems should as much as practical, adopt open architectures and support a collaborative development framework. Additionally, the open systems approach should support coalition partner efforts to develop unique toolsets of their own which they would integrate into there own systems.



Visualization: The visualization layer takes content (location of forces, readiness state, etc) and displays it in a usable fashion via a visual display. Some examples of visualization include map displays, tables, and graphs. A critical aspect for visualization is to apply the SOA decomposition of software into reusable segments of functionality. This means that we must separate the visualization service (the delivery content via a visual display) from the data management, data source or other services that actually generate the content. By keeping the visualization service separate from the computing intensive data management, we open up the possibility of using any number of visual displays – from a web-page based low-computing-capability to a three-dimensional rich client. In other words, we are not forced to use one device such as a high-performance desk-top machine that may be inappropriate for field tactical use. Additionally, separating the layers supports use of software applications like Google Earth or other available mapping services, to display content.

Collaboration: Collaboration software provides tools to communicate between users within the framework of the information system. In a C2 context, collaboration can be facilitated by instant messaging systems, chat, Voice over IP (VOIP) systems or other collaborative white boards which share C2 data. The quality of a common view of the battlespace is enhanced when warfighters working the picture can rapidly collaborate on the picture to resolve ambiguities and add value to the content. Although a seamless user experience is desired from the warfighter perspective, we are not advocating hard coding proprietary standards into the interfaces between the display systems, data management, and the collaborative software, rather we recommend employing open interface standards. Open standards such as extensible messaging posting protocol (XMPP) offer open methods that allow systems to collaborate by passing XML messages with standard ports and interfaces, supporting many SOA design principles. An example of a seamless user experience is a scenario where a warfighter could drag an object from the display and share that object with another user who may be using different display software over the collaborative system. The result is a C2 system with a seamless transfer of content between its collaboration, display and data management system that is not locked into a specific vendor solution.

Data Management: Data management provides a number of essential functions and executes much of the intensive computing – or “heavy lifting” – for a C2 system. The basic management activities to create, update, and delete are performed by the data manager as well as synchronization between partner systems to ensure a consistent picture. Since Coalition C2 software systems need to operate in limited bandwidth, high latency environments the data manager will additionally need to conduct advanced filtering to ensure only the desired data is sent over the network. Furthermore, the data manger will often employ some kind of compression technology to further reduce the network load.

Utilizing SOA technologies a Coalition C2 system should employ an agreed upon XML data definition and utilize a common messaging such as SOAP, JMS or ReST technology to support interoperability. The more standardized our data definitions are the better our computer systems will interoperate, however just like we should not wait for all the standards to mature before building a coalition C2 capability we should not wait for a universally approved XML data definition from a joint or international body, before deploying a solution. The goal of a coalition based C2 computer/network system is to provide an economical, scaleable data management and synchronization service that is interoperable with a wide variety of visualization applications and persistence systems. The challenge remains balancing the demand for proper system configuration control and interoperability with the freedom to solve coalition partner's unique needs. In the next section we will discuss software patterns that provide guidance in the development of coalition C2 systems and detail potential acquisition strategies.

Trickle-up and *Zone* Patterns

Software patterns and data strategies provide developer guidance to ensure broad system interoperability. This article discusses two patterns, the first, *trickle-up* provides guidance on developing a broad data source infrastructure, the second, *zone* pattern, details how a COP application interacts with its sources, partners and clients. When combined, these two patterns provide a new approach on developing a scalable and flexible Coalition C2 system built on web-services. Figure 3 shows the trickle-up pattern three levels; reports, tracks and entities. The pattern has two general types of software artifacts; object manager services (OMS) and data fusion services (DFS). An OMS is a web-based data repository that performs the initial transform from the sensor or other source and stores the content for follow on query. The DFS is a correlation or fusion agent, which takes content from one or more OMS and combines it to create value added data. In the pattern, the OMS resides in the three layers while the DFS reside between the layers. Using a brick and mortar analogy, the OMS is a brick and the DFS is the cement binding the bricks together.

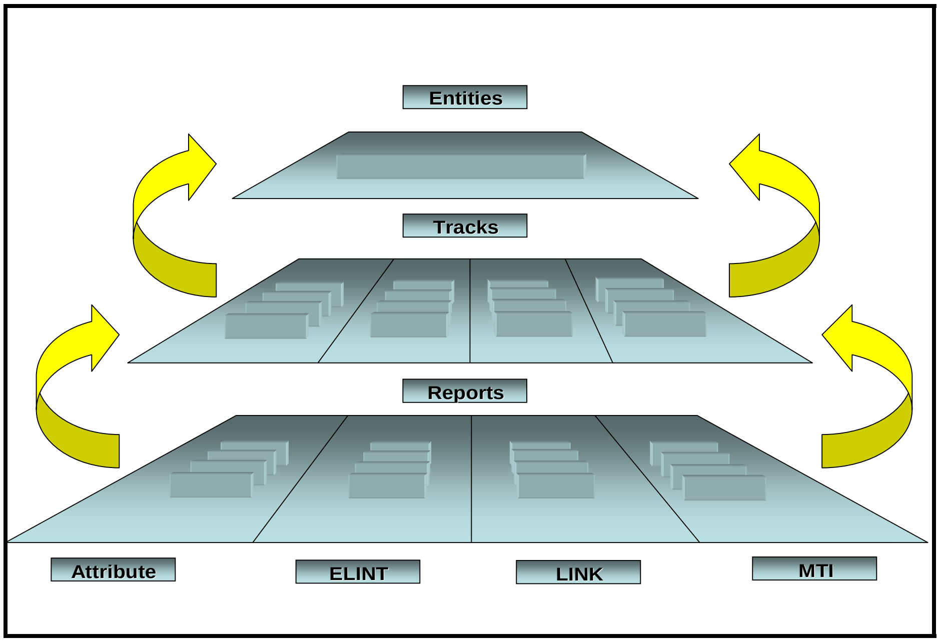


Figure 3. SOA Three Tier Data Model

The *trickle-up* pattern was initially proposed by eXtensible Tactical C4I framework team at the San Diego, Space and Naval Warfare Center to describe a new paradigm for management of sensor data, fusion algorithms and its interaction with the COP (XTCF, 2). The pattern provides a basic partitioning of the data into three levels creating ontology by data complexity to enable source discovery. The goal of a framework developed from the pattern is to lower the barrier of entry for new services to enter the domain and allow dynamic employment of new fusion and correlation services. The model is divided into three levels, reports on the bottom, tracks in the middle and entities on top. The pattern has a loose mapping to the Joint Directors of Laboratories (JDL) data fusion model (Steinberg, 2-4). The JDL data fusion model provides a functional view where data is categorized by the complexity of the role it plays, while the *trickle-up* pattern in contrast is a simpler model that categorizes content by the complexity of the data object without reference to the role. The *trickle-up* pattern has three general purposes in the information-exchange domain. Firstly, it articulates that each data source is its own unique service separate from a service that may fuse or correlate the data source. Secondly the model provides a basic ontology structure for the data services to provide tractability as data sources are fused with others to create new source services. Thirdly the model provides a view of how the services are organized to allow discovery services to discriminate them.

As discussed previously, the model has three levels: reports, tracks, and entities. Reports are the lowest level of data service and covers atomic sources of content, which are usually raw data sources. Reports are generally facts that alone provide limited information from a single source. Reports include data objects such as: acoustic reports, electrical intelligence (ELINT) intercepts, Moving Target Indicator (MTI), link or combat system interface data (TADIL) or other atomic type data sources. For example, an ELINT report would contain details such as a frequency and line of bearing or and acoustic report has a frequency or harmonic data. Reports are usually limited and only begin to reveal higher order information when they are aggregated with other reports or combined with data from other devices.

The next level, tracks, provides a view of more complex objects, built upon the atomic content provided by reports. Tracks often have position and direction as a function of time associated with them. For example, a track may be comprised of a series of ELINT reports that over time reveal the movement and hence the course and speed of an object. Track objects are also formed from dissimilar reports by means of data fusion. For example, an ELINT report which reveals an initial position can be augmented by an acoustic report which further elaborates on the type and intended movement of the object and hence become a track. This dissimilar report aggregation and promotion to track relies on a fusion engine or DFS from the previous section to fuse the reports. This discussion, introduces another potential power of the flexible SOA system--the ability to integrate at runtime a different fusion “services” customized for the specific tactical environment.

The highest level of data object in the trickle-up pattern is the entity. Webster’s dictionary defines an entity as “something that has separate and distinct existence and objective or conceptual reality.” For the purposes of this paper, entities refer to data objects that are combined with non-kinematics data such as a “cargo manifest” or adversaries historical patterns. Additionally, entity may contain a target folder, with routes, battle-space geometries/overlays and other associated objects identified along common lines.

The *trickle-up* pattern provides a structure to organize data sources and fusion logic in a coalition military framework. Partitioning the sources not only allows us to create a *plug and fight* paradigm for our fusion and correlation logic, but allows us to organize our content for more dynamic discovery. Next we will examine a model that provides a structure for an application to consume the content.

Command and Control Zone Models

Section one describes the emerging focus on Navy missions in stability operations, global war on terrorism, and homeland defense. These missions drive a different type of coalition Common Operations Picture (COP) topology than those in traditional major combat operations. Major combat operations, such as blue-water naval carrier and expeditionary strike group (CSG/ESG) operations, COP topologies tended to follow a rigid synchronization model. This *top-cop* moved data up the hierarchy to a central master database, which then pushed a common picture down the tree in an attempt to synchronize the data among all nodes. This hierarchical *top-cop* topology supported unity of effort and a common view of the battlespace; however it wastes bandwidth and computing cycles as it moves data to every node whether it's needed or not. Additionally, a hierarchical COP dilutes the sense of purpose and mission as every warfare mission's content (e.g. air, surface, ASW) is mixed into a *top-cop* database. Alternatively, an *ad-hoc* type COP topology allows the commander to tailor the COP for a specific mission by only sharing content worth promoting and focusing exclusively on content required to support the mission. These concepts translate into software with two general requirements; data flexibility and ad-hoc partnering. Figure 4 shows the general difference between the hierarchical and distributed COP topology.

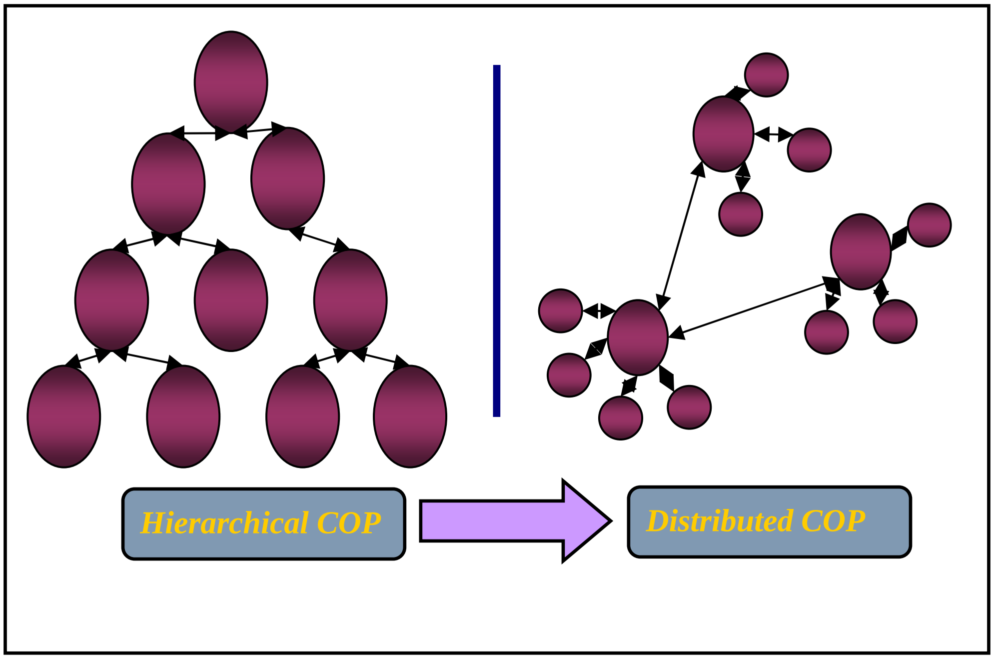


Figure 4. Hierarchical to Distributed COP

Operational flexibility and ad-hoc partnering requirements are enabled by SOA technology, however doctrine, tactics and procedures need updating to effectively employ these technologies. Data flexibility for COP software systems can be measured by the ease associated with adding and removing sources of content such as a new data feed or in the case of this paper the ability to reach into the *trickle-up* pattern of reports, tracks or entities. Past COP systems where tightly coupled to the sources of data they managed. A web-services based SOA COP system employs standard data definition and transport technologies supporting broad interoperability goals. The second component is *ad-hoc* partnering, which is the concept of rapidly establishing, updating and deleting COP data sharing relationships. In some ways, three general roles exist in a common operations picture, which map to the relationships in a tactical data link system. These roles are TADIL-COP leader, participant, and consumer (Logicon, 40). Expanding on the TADIL-COP roles, the diagram below is an operational view of a zone pattern COP topology, where each zone or community of interest, establishes and maintains its own COP, with participants, data sources and consumers; described in greater detail below.

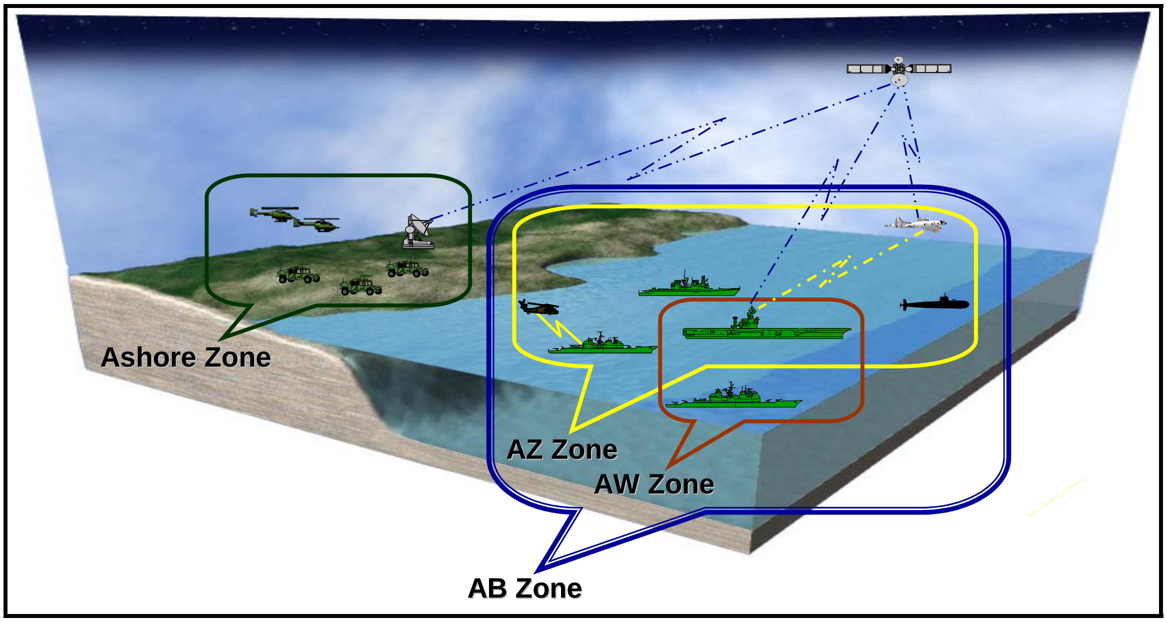


Figure 5. Zone Distributed COP

Using the previously described patterns, a SOA C2 Architecture has two main components, the infrastructure source components described by the *trickle-up* pattern, and data management components described by a zone pattern distributed client model. A role of the zone is to provide a mechanism by which a user can manage system interfaces with the data source infrastructure described in the *trickle-up* pattern. The zone pattern manages views and in other ways employs the data in the infrastructure layer. In much the same way a web browser utilizes the data from the Internet, the zone pattern interacts with the trickle-up infrastructure. In figure 6, we apply the zone pattern to a reference composite warfare commander (CWC) doctrine of multiple warfare areas reporting to a common commander. In the below framework, AB is a carrier strike group commander, AZ is a coalition anti-submarine and anti-surface warfare commander and AE is the electronic warfare commander. Here we explore three relationships described the zone pattern: zone-to-client, zone-to-zone and zone–to-track or data feed. The zone pattern reflects the three intertwined "services" that the model must manage. Expanding the definitions further we have: the relationship between the zone and the data sources of trickle-up sources, the relationship between the zone and other users who participate with the owner of the zone, and the relationship between the zone or owner of the data and the consumer of the COP. The second and third relationship parallels a magazine printer relationship with its fellow writers and readers, in that the content available to those who edit the magazine is different than the finished product provided to subscribers. In a zone COP system, the “raw” view that action officers are working on is different than the one provided to a commander or senior decision maker.

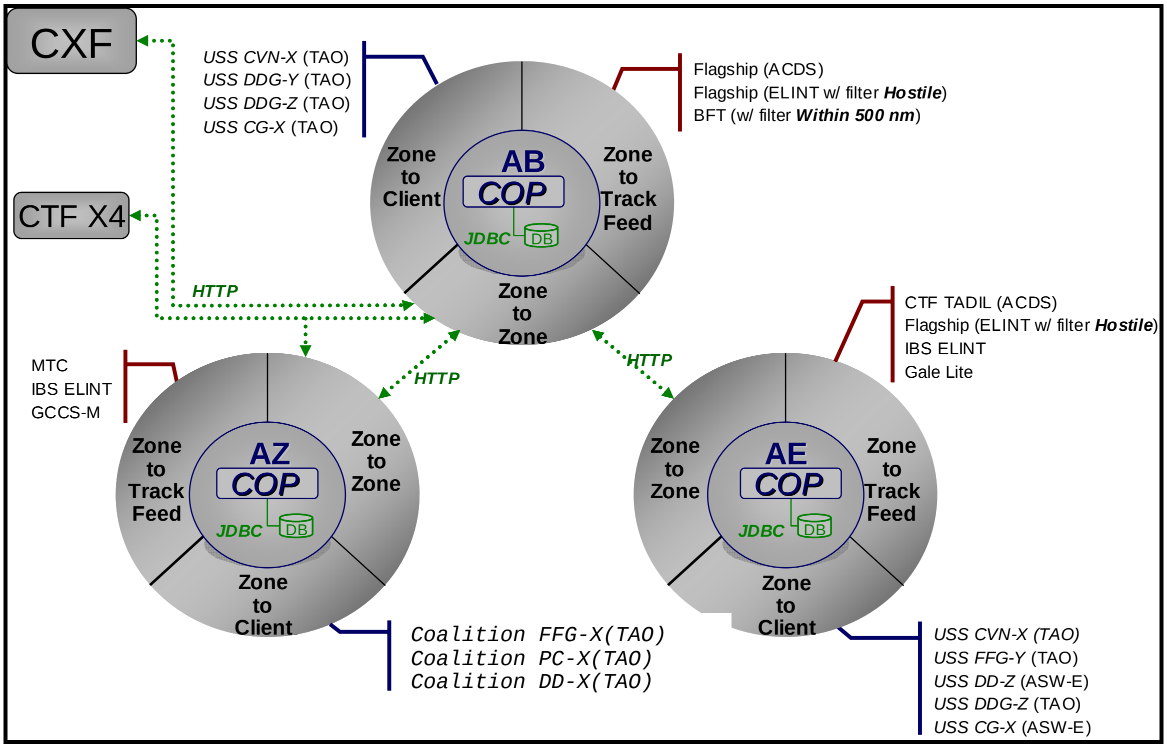


Figure 5. C2 Zone Data Model applied in the CWC Doctrine

Examining figure 5, we have three zones, AB – the overall commander, AZ – the sea combat commander (who could be a coalition partner) and AE – the electronic warfare commander. Note that each has sources, clients and consumers. In this example, AB consumes AZ and AE COP, while also subscribing to sources of its own. Additionally, using software filters the consumer of the COP can specify the desired content, as well as allowing the producer to set which content they would like to share. A difference between the COP shared by participants of a zone and those who consume it is a central requirement of the zone pattern that fosters accountability and scalability in the COP. Accountability is supported by zone ownership, where only the zone leader can approve object deletions and modification to the COP structure, and hence is held accountable for its quality. The zone pattern supports scalability by creating new zones or COP’s as size and complexity of an existing zone increases. Similar to biological cells, a collection of smaller zones or cells create a larger organism or a larger zone. In the next section we will discuss the development framework to build coalition C2 systems in a collaborative manner.

Coalition software development

Software engineers often stress the requirements phase as one of the most critical of the software lifecycle (Pressman, 272). This remains true with SOA technologies. Theoretically, SOA provides a rapid development and integration cycle, however a complete understanding of the problem remains essential to delivering a product users need. In the case of building a C2 software system for a coalition thousand ship navy we need to target those requirements that drive a successful coalition in the types of activities they normally conduct such as stability, GWOT and some unique cases homeland defense vice major combat operations. Coalition warfare is a dynamic environment that requires flexible software systems. Command relationships change often and must be easily established and changed (Alberts , 2). Additionally, relationships in the coalition environment are time dependent and temporary, and any future C2 system must allow the user to quickly shift from non-participant observer, to participant, to leader, based on changing tactical environment. Furthermore, data source flexibility is essential. Data source flexibility means not sending out a new build of software when a new data sources are added. Web based SOA's have the potential to deliver on required flexibility; however a third requirement speaks to how these systems are engineering from the ground-up.

A third requirement to develop a multinational system is multinational development. The United States is often a leader in advanced technology, especially in weapon system development. The paradigm for foreign military sales is often based on sales of existing U.S. systems with some modification of the system to suit local requirements. Buying and modifying U.S. systems might work well for systems requiring large capital investments such as aircraft, ships or other hardware intensive systems. However, this may not be the best method to develop and field a software system to share data in a coalition environment. Successful complex software systems have been developed in a more international collaborative style, such as the IBM sponsored Eclipse Framework or the Apache web server. In these cases a limited group of developers built an initial capability and then opened the design details and source code to a wide audience for improvement and use. This model resulted in an improved system with wide adoption and market penetration. The key to success was firstly an early viable system, and secondly the content was hosted and provided in a manner that facilitated quick understanding and collaboration. Those keys will be essential to developing a suitable multinational development environment for coalition COP systems and is the approach advocated in this article.

A Modified Open Source Environment Approach

Military system development is not identical to commercial systems. Requirements such as security, availability and reliability are often even more critical in a military system than a commercial one. In light of these limitations, a purely open source development model is not suitable for coalition Common Operations Picture development. Instead, limited collaboration aligned to the various coalition network domains strikes a good balance between security and open-development. For example, existing secure networks such as those between the U.S, Australia, England, New Zealand, Korea and Japan can provide the interoperability and security required to develop between these coalition partners.   This potentially would be an ideal environment to host a development website where the executables, design documents, installation instructions and source code from the initially developed SOA C2 capability would be posted. Beyond the benefits normally associated with collaborative development, this site would build trust between the US and partner nations. Trust, where allies know what technical direction the US is going and trust that C2 investments made will not be wasted. Figure 6 below shows a sample of what that system may look like. This web interface hosts the source code, executables, COTS infrastructure, design and user manuals, but more importantly provides the tool to validate a new system still interoperates with the existing system, by providing a live-test site. Additionally, general configuration management and simple collaboration services are also available to the authorized developer.

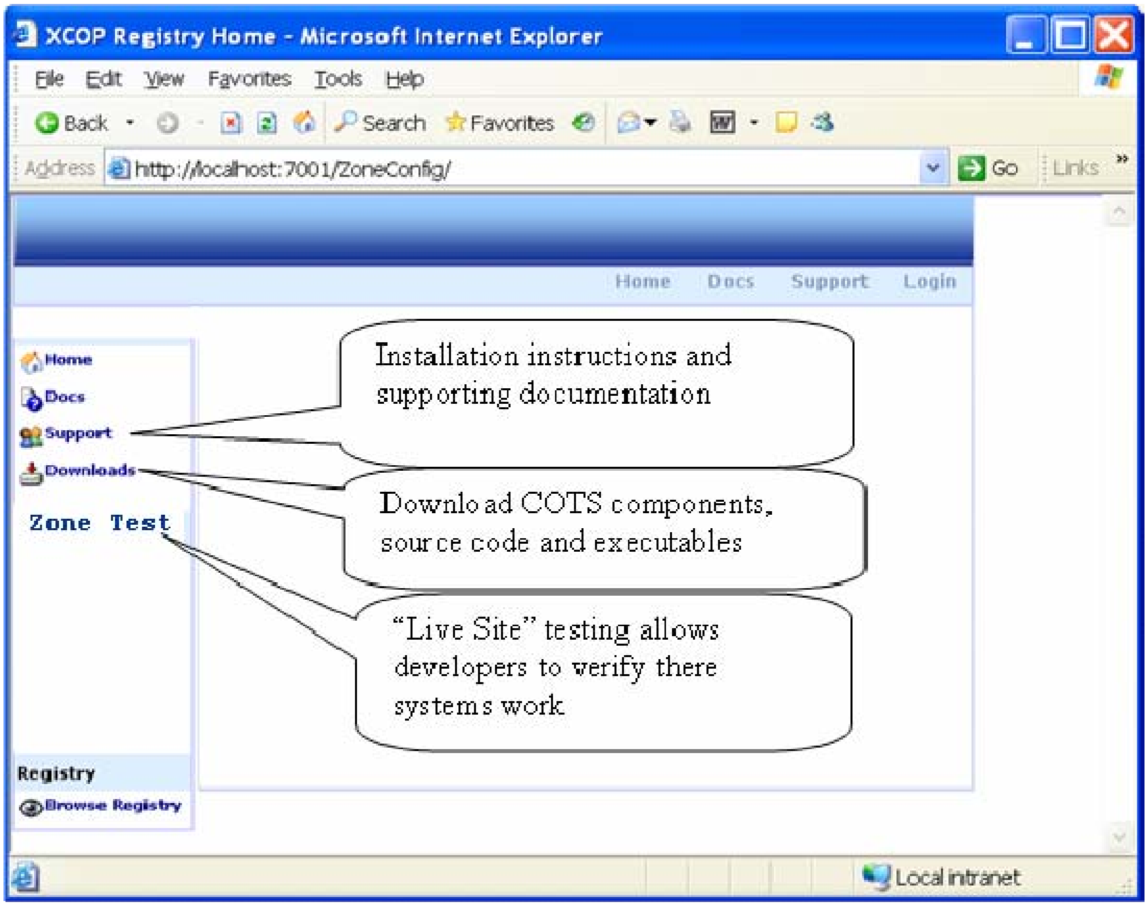


Figure 6. Sample Coalition C2 System Development Web-Site

Conclusion

SOA has potential to improve a coalition of navies' ability to rapidly integrate new data sources and user participants into a common view of the battlespace (e.g. provide increased flexibility) with potential resources saved by leveraging industry investment in IT technologies. SOA based capabilities have immediate applicability in operations other than major combat operations involving stability operations and GWOT mission areas, where collaboration with other navies is critical. These capabilities foster the required flexibility and ad-hoc command structures necessary to support coalition warfare. SOA technology alone however, will not deliver the needed system interoperability. Beyond a cultural shift in how we share data and changes in tactics and procedures we must change how we design, build and acquire coalition C2 systems. A collaborative development environment following established industry methods is additionally essential. A collaborative web-based method allowing participation and hands-on development by coalition partners and hosted by the U.S. government has potential to rapidly deliver a C2 system to operate in a coalition environment. Delivering these tools requires a change in methods. It involves the acquisition community to not merely take requirements from the warfighter and attempt to churn them into product, but rather to take risks, to develop trust in its coalition partners, and in some cases to adjust processes to operate in a global development environment.

LIST OF REFERENCES.

1. Steinberg, A. N., Bowman, C.L., “Revisions to the JDL Data Fusion Model”, *Handbook of Multisensor Data Fusion*, CRC Press, Boca Raton, 2001.

2. Alberts, D.S., “Coalition Command and Control: Peace Operations,” *Strategic Forum Number 10*, October 1994.

3 Chief of Naval Operations Guidance for 2006, http://www.navy.mil/features/2006CNOG.pdf

3. Erl, T., “Service Oriented Architecture Concepts, Technology, and Design”, *Prentice Hall*, New York, July 2005

3. Pressman, R. S., “Software Engineering, A Practitioner’s Approach Fifth Edition”, *McGraw-Hill*, New York, 2001

4. Space and Naval Warfare Systems Center, San Diego (SSC-SD). Developer Guide for the Extensible Tactical C4I Framework (XTCF) Phase 3.2 Software Development Kit. 23 April 2004.

5. Thomas P.M. Barnett, “The Pentagons New Map”, *Naval Institute Press*, Newport RI, 2002

6. Graham, S., “Building Web Services with Java”, *Sams Publishing*, Indianapolis, Indiana, 2005

7. Department of Defense Chief Information Officer (CIO), Department of Defense Net-Centric Data Strategy, http://www.defenselink.mil/cio-nii/docs/Net-Centric-Data-Strategy-2003-05-092.pdf